

THE ULTRASWITCH

- a remoteqth project by DM5XX -



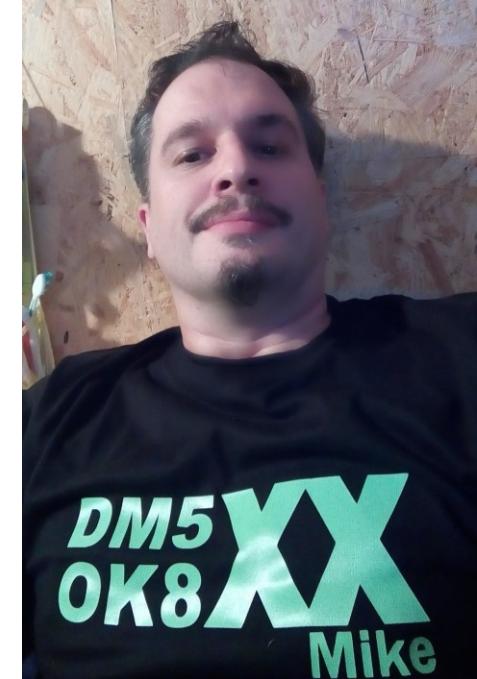
remoteQTH.com

- OPEN-SOURCE Hardware + OPEN-SOURCE Firmware
- OK1HRA, Dan | OK2ZAW, Jan | OK1CDJ, Ondra | DM5XX, Mike
- <https://remoteqth.com> + qro.cz + hamshop.cz = [OL7M](#)
- K9AY
- Beverages: Top band, single band, bi-directional 2-wire, 4-way bi-directional,...
- Switching:
6To2/6To1/4To1, 6To2+Triplexer, 1To2/1To3 Stackmatch, 12x4,...
- Variable ATT, SO2R, M/M, Controller (on-site/remote), Custom Contest Hardware...



DM5XX

- Funkamateur seit 1990 (OK8XX, DM5X, ex. DG3NED, ex. DL3NED) aus Nürnberg
- Senior Software Engineer
- Derzeit IT-Projektleiter (Leitung Niederlassung) @ ADITUS GmbH
- Seit 2015 Lead Firmware Developer @remoteqth.com
- Hardwarenahe Microcontrollerprogrammierung



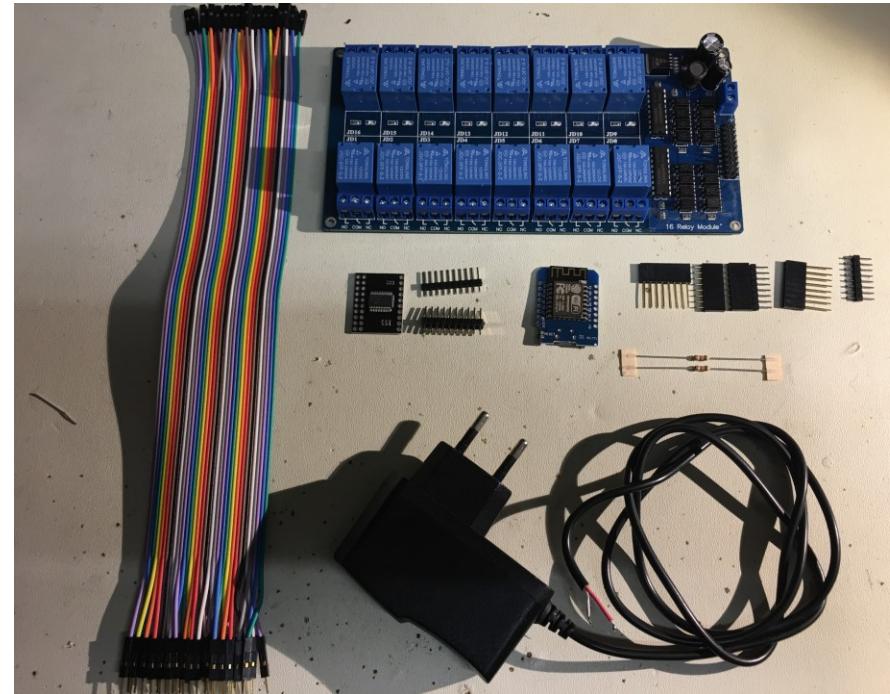
The Ultraswitch

- Wifi basiert & inspiriert vom [mega-webswitch](#) (63-CH remote-switch)
- Teil des 256-Port-Inband-Antennenprojekts @ OL7M
- Einfache Hardware direkt aus China (ebay.com)
- ESP8266-12F („Wemos D1 mini“)
- Modular: Bis zu 8 Boards
- Web-UI & Web-Config
- RESTful und einfach erweiterbar dank web-basierter Architektur



Die Hardwarekomponenten

- 16-CH Relayboard mit 12V-Spannungsversorgung
- MCP23017SMD auf PCB
- Wemos D1 mini
- 2x 1.8K
- DuPont Steckverbinder
- 12V-Netzteil
- Computer+MicroUsb-Kabel (Progr.)



Die Architektur

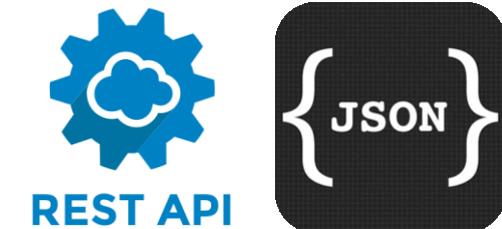
Separation of Concerns:

- Hardwareschicht (1)
- Hardwarenahe Funktionalität (JSON) (1)
- Funktionensschicht (JS) (2)
- Anzeigeschicht (UI) (3)

=> aka 3-Layer-Modell

Browser:
(3) Anzeige

Javascript:
(2) Steuerfunktionen



(1) ESP12F/RelayBoard



Vor- und Nachteile

Vorteile:

- Leicht erweiterbar, Unabhängig von der UI
(Web-App, Windows-App, Mobile-App, Linux-App,...)
- Trennung von Hardware, Logik und Funktionen
- Zentralisiert, Modular, Webtechnologie, Zukunftsfähig
- Eigene Funktionen und UI (Customizing)

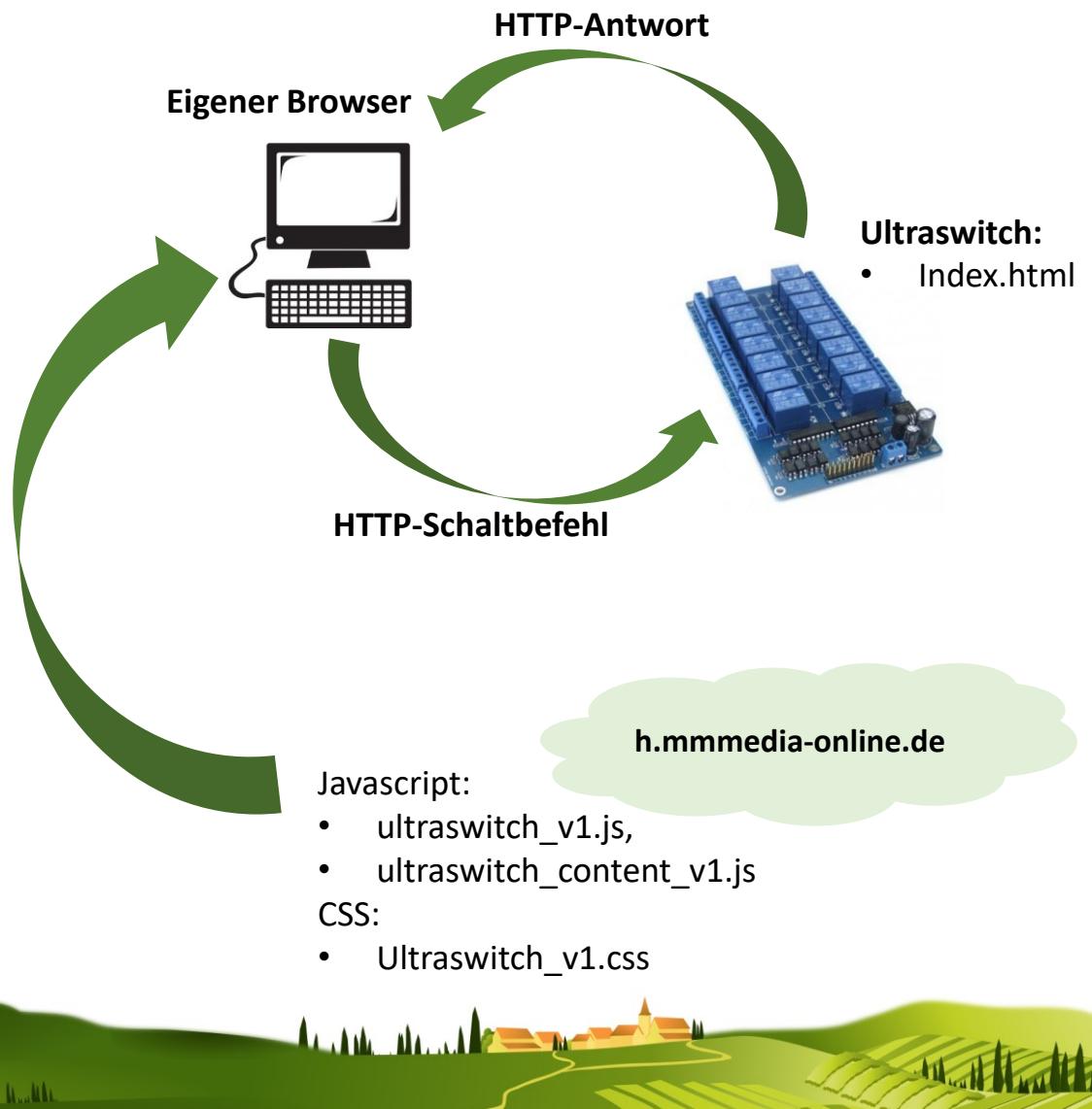
Nachteile:

- Sicherheitsmechanismen liegen in eigener Verantwortung der Funktionsschicht
- „Viel Feind, viel Ehr“ (Je mehr angepasst wird, umso mehr Wissen ist notwendig)



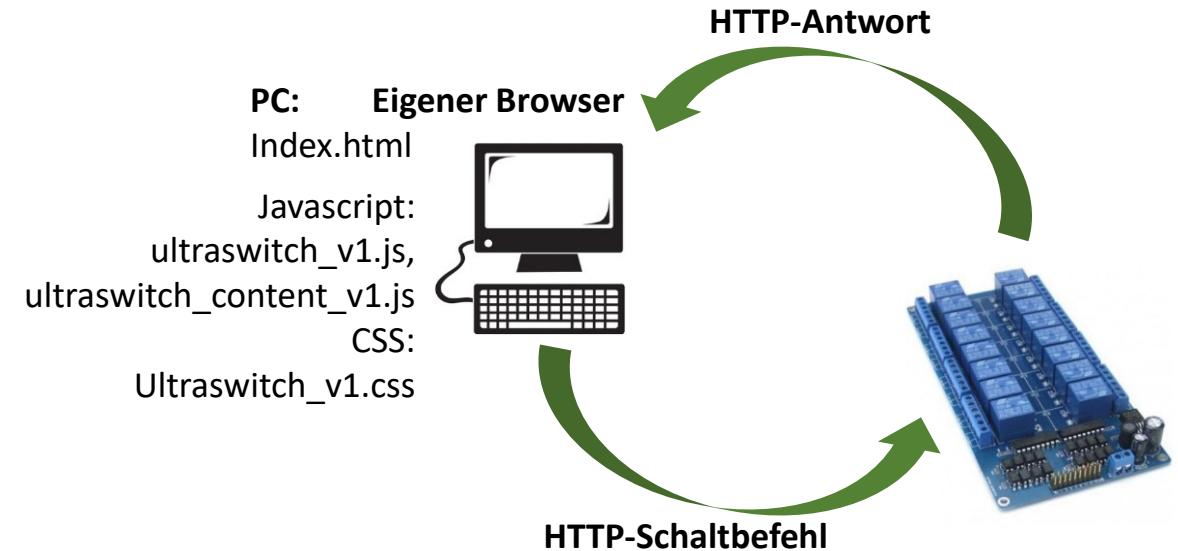
Usecase 1 – „Out of the box“

- Keine Anpassungen notwendig
- Alle vorbereiteten Ressourcen („h.mmmmedia-online.de“) werden einfach verwendet und genutzt
- Nur Konfiguration des Switches notwendig (IP-Adresse vergeben)
- Version immer up-to-Date
- Nachteil: Beschriftung kann nicht geändert werden.



Usecase 2 – „Eigene Desktop-Anwendung“

- Anpassungen einfach möglich
- Alle Ressourcen lokal auf dem PC
- Alles anpassbar
- Kein eigener Webserver notwendig
- Nachteil: Keine automatischen Updates, nicht mehr Multi-User-Fähig (Logik/Infos liegen auf dem eigenen Rechner).
Nicht mobile-fähig.

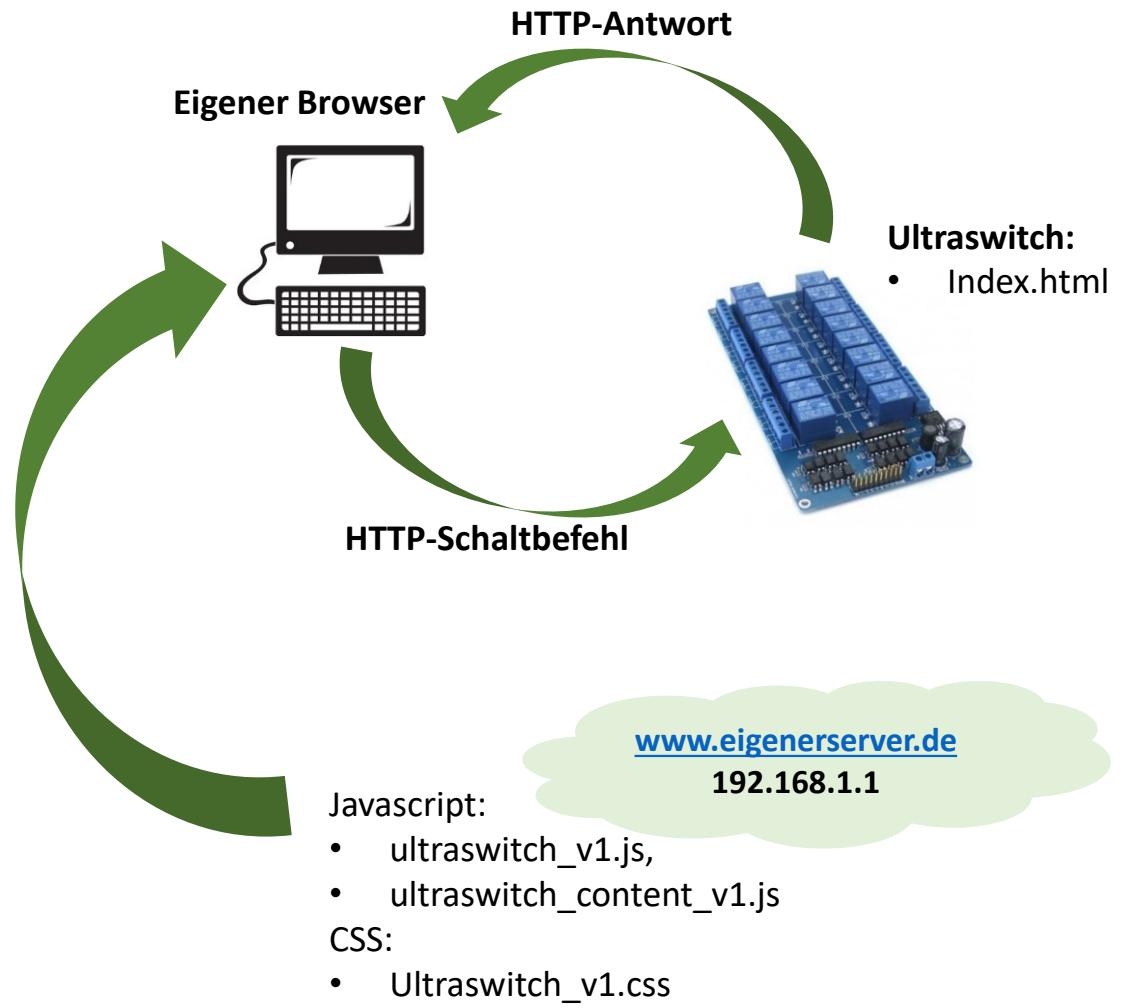


Usecase 3 – „Eigener Webserver“

- Vereint Vorteile aus 1 uns 2
- Multi-User-Fähig
- Flexibler Zugriff
- Server kann im eigenen Netz oder extern Sein

Nachteil:

- Webspace notwendig
- WebServer-Kenntnisse notwendig
- Ebenfalls Keine automat. Updates



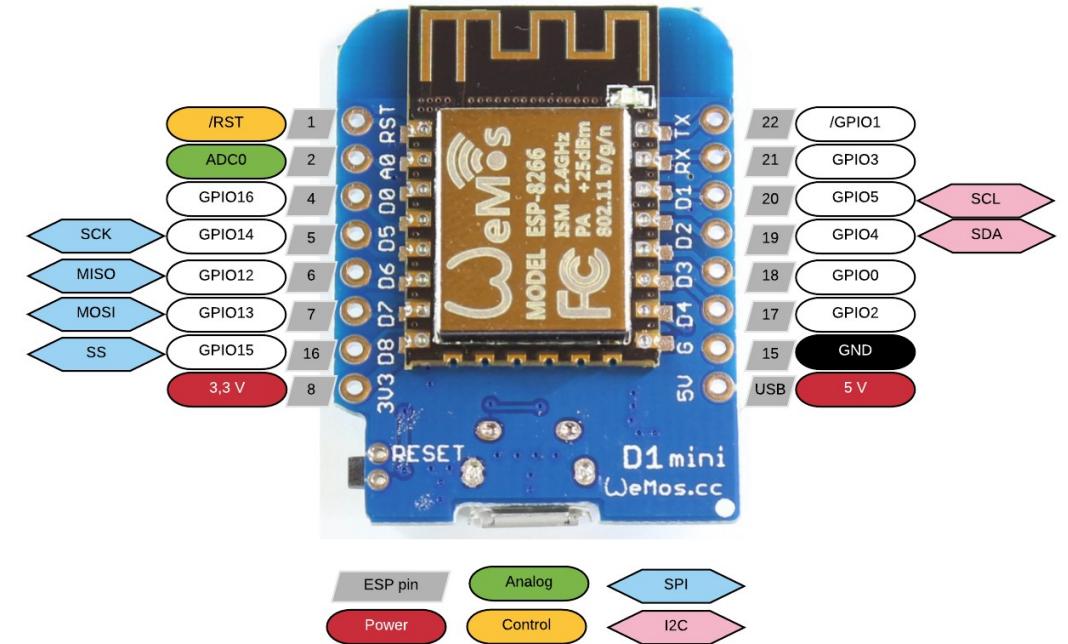
Good to know

- Einfache Erklärung „[Netzwerkgrundlagen](#)“
- Einführung in „[Übertragungsprotokolle](#)“
- [JSON](#)
- [HTML, CSS, Javascript](#)
- Introduction into [REST](#)



Die Hardwareplattform – ESP8266-12F

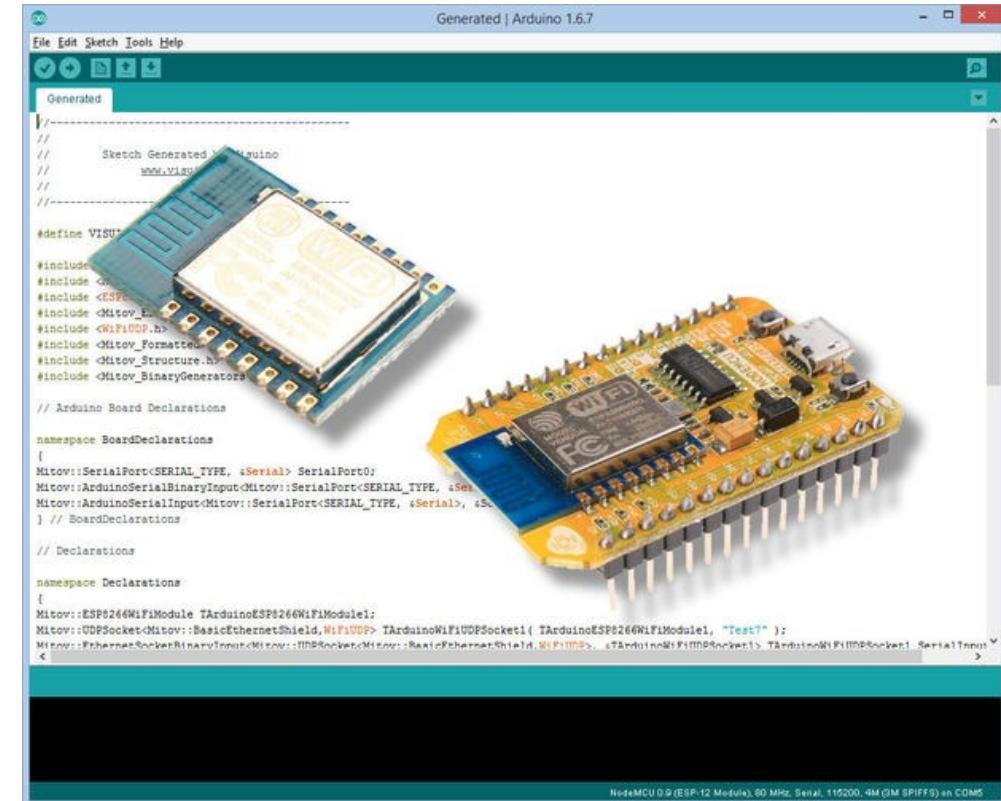
- 802.11 b/g/n, Wi-Fi 2.4 GHz, support WPA/WPA2
- Integrated low power 32-bit MCU @80mhz, 4MB Ram, 92kb
- Integrated 10bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA,
- power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO, STBC, 1x1 MIMO, 2x1 MIMO
- Deep sleep power <10uA, Power down leakage current < 5uA
- Standby power consumption of < 1.0mW
- <http://stefanfrings.de/esp8266/>
- [Datasheet](#)



Arduino-IDE

- <http://www.gunook.com/einrichten-der-arduino-ide-zum-programmieren-von-esp8266/>
- http://arduino.esp8266.com/versions/2.3.0/package_esp8266com_index.json

(Achtung: 2.3.0! 2.4.0 hat einen Bug)



Kurzer Ausflug in die IDE und Programmierung

- <https://www.youtube.com/watch?v=0wAY3DYihyg>
- <https://www.youtube.com/watch?v=BKm1QFZS0m4>

- <https://www.youtube.com/watch?v=eaFvQG8wrGw&list=PLAB63281B90FB376E&index=3>
- <https://www.youtube.com/watch?v=t-lRfOalxhE&index=4&list=PLAB63281B90FB376E>
- <https://www.youtube.com/watch?v=nf8Uqy6vLmg&index=5&list=PLAB63281B90FB376E>
- <https://www.youtube.com/watch?v=QIDH2SApocM&list=PLAB63281B90FB376E&index=6>



Libraries (Bibliotheken und der Sketch)

- [Artikel zu Libraries](#) (heise.de)““

Nach der IDE-Installation liegen die Projektfiles...

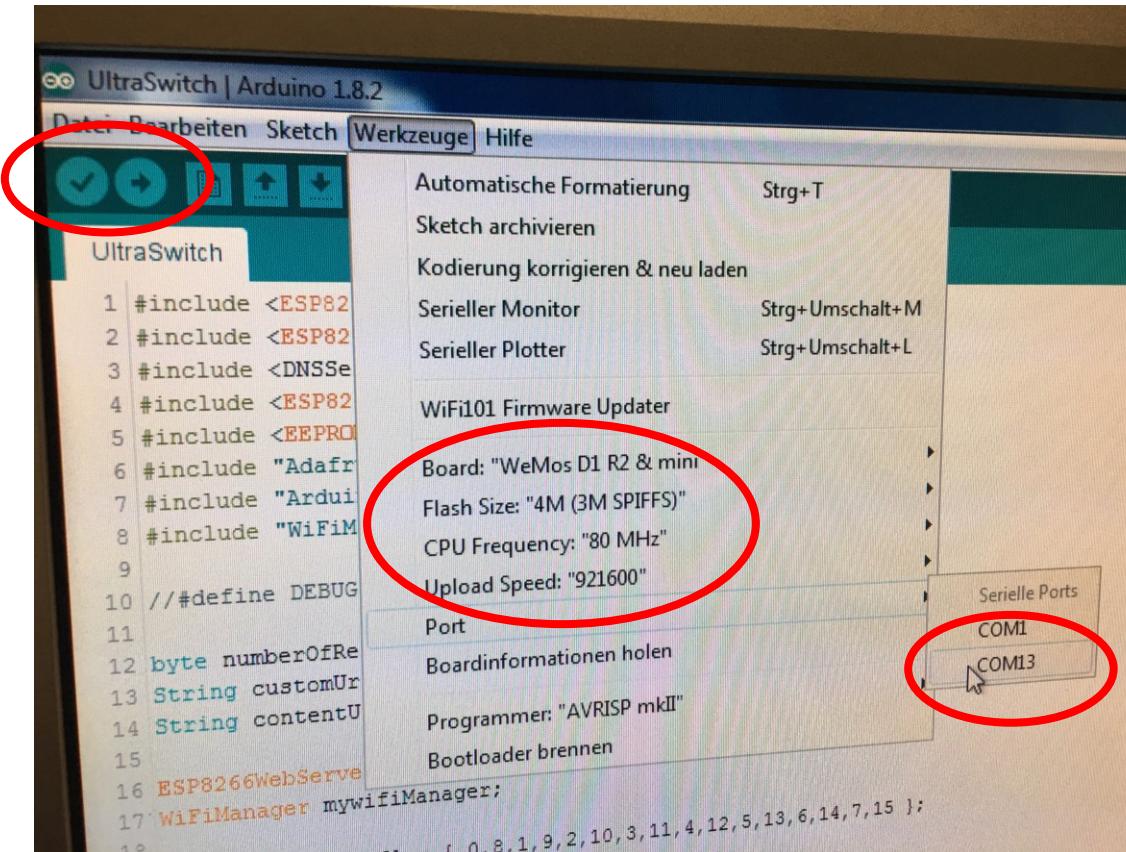
- C:\Users\mmmedia\Documents\Arduino
=> Folder Ultraswitch dorthin kopieren
- C:\Users\mmmedia\Documents\Arduino\libraries
=> Folder aus “Ultraswitch/needed_libraries” hinkopieren
- Auf den Desktop kopieren: (für später)
C:\Users\mmmedia\Documents\Arduino\Arrest\ultraSwitch_web



Los geht's: 8266-12F

- Board ESP hinzugefügt?
- Libraries kopiert?
- Sketch geladen?
- Einstellungen angepasst?
- Serieller Port stimmt?
- 12F mit USB verbunden?

JA? => Verify > Hochladen!

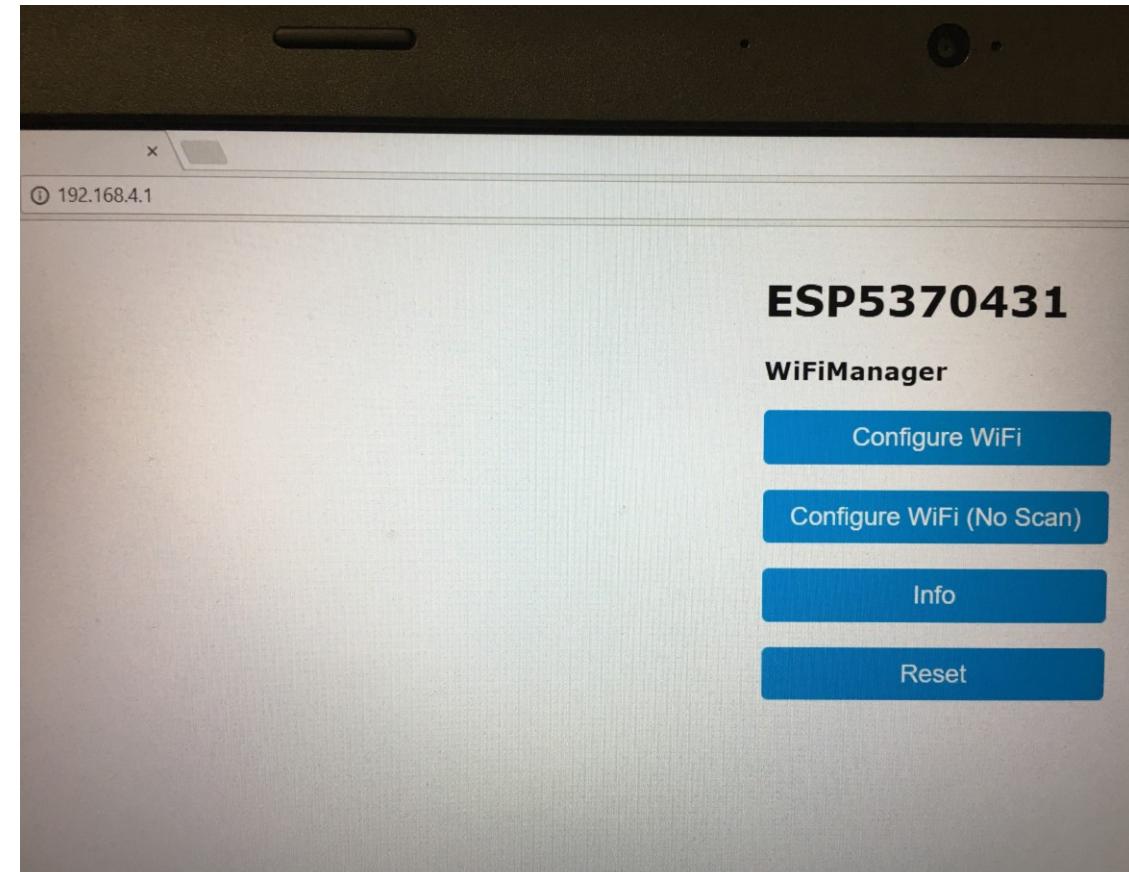


Mit dem 12F verbinden – AP-Mode 1

- Wifi-Netz ESP<Nummer> suchen
- Mit ESP<Nummer> verbinden
- Browser öffnen (Default: 192.168.4.1) – nichts weiter tun!

Achtung RRDXA-Workshop:

Bitte immer nur 1 ESP angesteckt haben! Sonst gibt es zu ESP-Wlans...
Aber welcher ist Eurer...?!



Mit dem 12F verbinden – AP-Mode 2

- Bitte setup_esp.html aus dem Ordner „Ultraswitch_Web“ mit doppelklick öffnen.
- Bitte Daten gemäß Eurer Info eintragen
- Aus „Submit“ klicken

The screenshot shows a web browser window titled "CONFIG UltraSwitch". The address bar displays "file:///C:/Users/mmmedia/Desktop/ultraSwitch_web/setup_esp.html". The main content area is titled "Welcome to the quick n dirty configurator for the ESP_Accesspoint!". It contains five input fields labeled (0) through (5).

- (0) Current IP-Adress of the AP e.g. 192.168.4.1 (no spaces, "-"-separated) --->
- (1) IP-Adress of the Controller e.g. 192.168.1.111 (no spaces, "-"-separated) --->
- (2) IP-Adress of your Internet-Gateway(router) e.g. 192.168.1.44 (no spaces, "-"-separated) --->
- (3) IP-Adress SubNetMask 255.255.255.0 --->
- (4) SSID --->
- (5) WLANPWD

At the bottom of the form, the text "BE CAREFUL WHAT YOU DO!! DOUBLECHECK TWICE BEFORE CLICKING SUBMIT :P" is displayed. A "Submit" button is located at the bottom left of the form area.

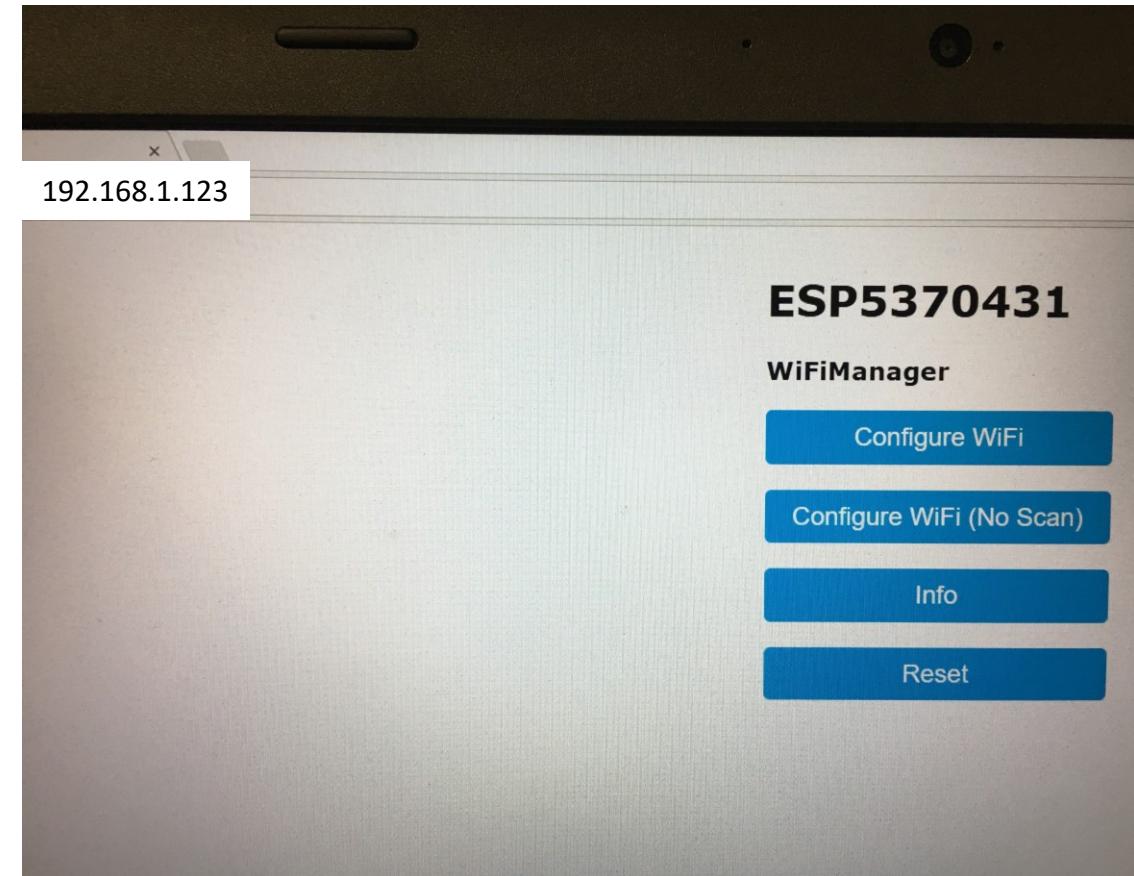


Mit dem 12F verbinden – AP-Mode 3

- Ins richtige WLan wechseln – nicht mehr ins ESP-Wlan! (jetzt wird's tricky)
- Browser öffnen (z.B. 192.168.1.123)
- „Configure Wifi“

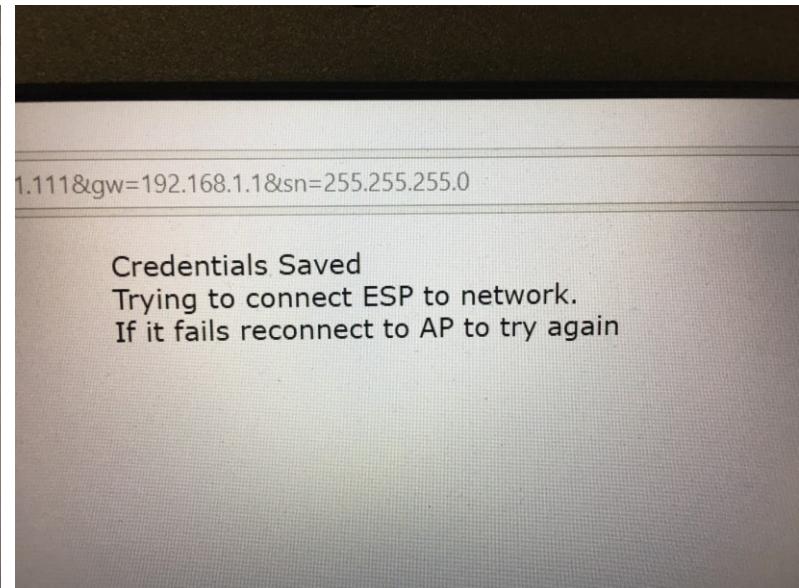
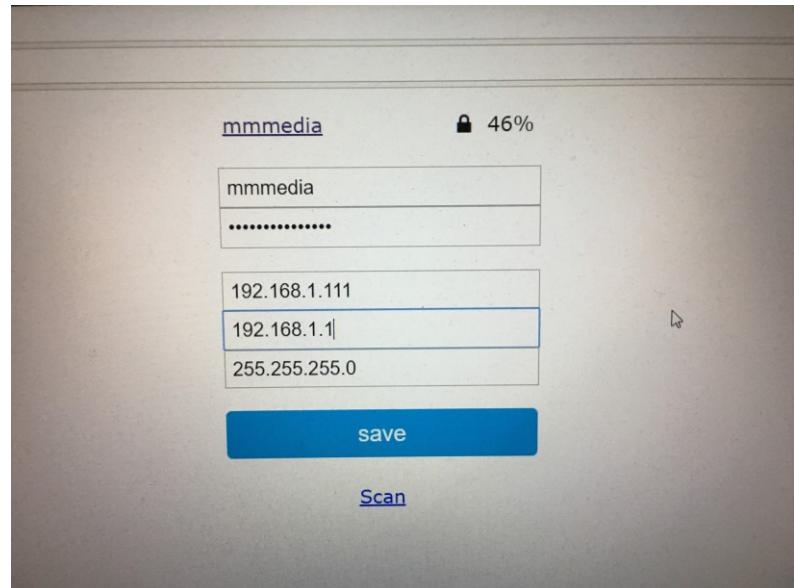
Achtung RRDXA-Workshop:

192.168.1.123 ist die zuvor vergebene IP-Adresse, unter der Euer Ultraswitch erreichbar ist.((o)))



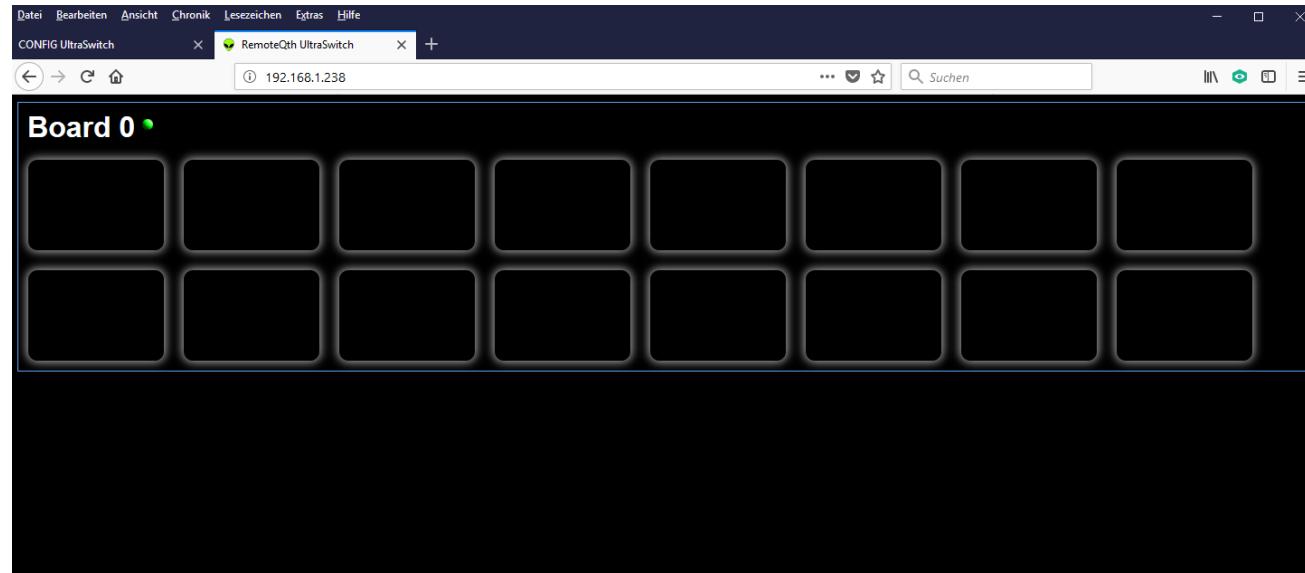
Mit dem 12F verbinden – AP-Mode 4

- „Configure Wifi“:
Daten SSID und PWD
nochmals eingeben!
- Daten unten prüfen!



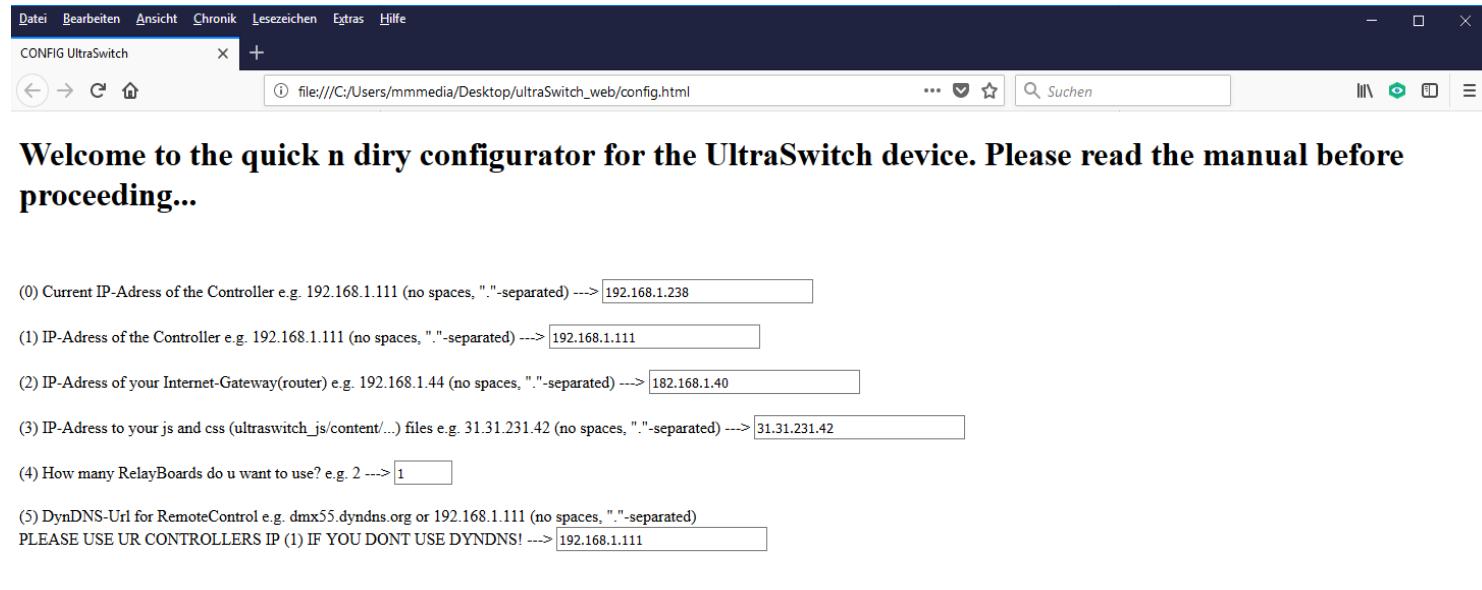
Mit dem 12F verbinden – Der erste Kontakt

- Die IP Eures Ultraswitch im Browser aufrufen...
- Troubleshooting: Ultraswitch nicht gefunden?
[Advanced IP-Scanner](#) hilft!
IP merken und im Browser eingeben!
- Aber Pfade stimmen noch nicht
=> Keine Buttons, etc...

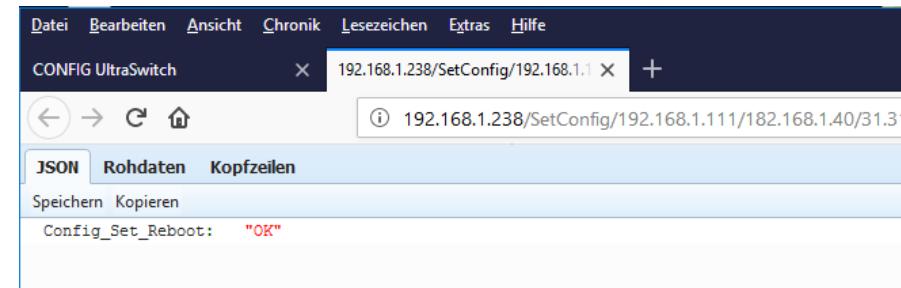


Mit dem 12F verbinden – IP-Settings Servermode

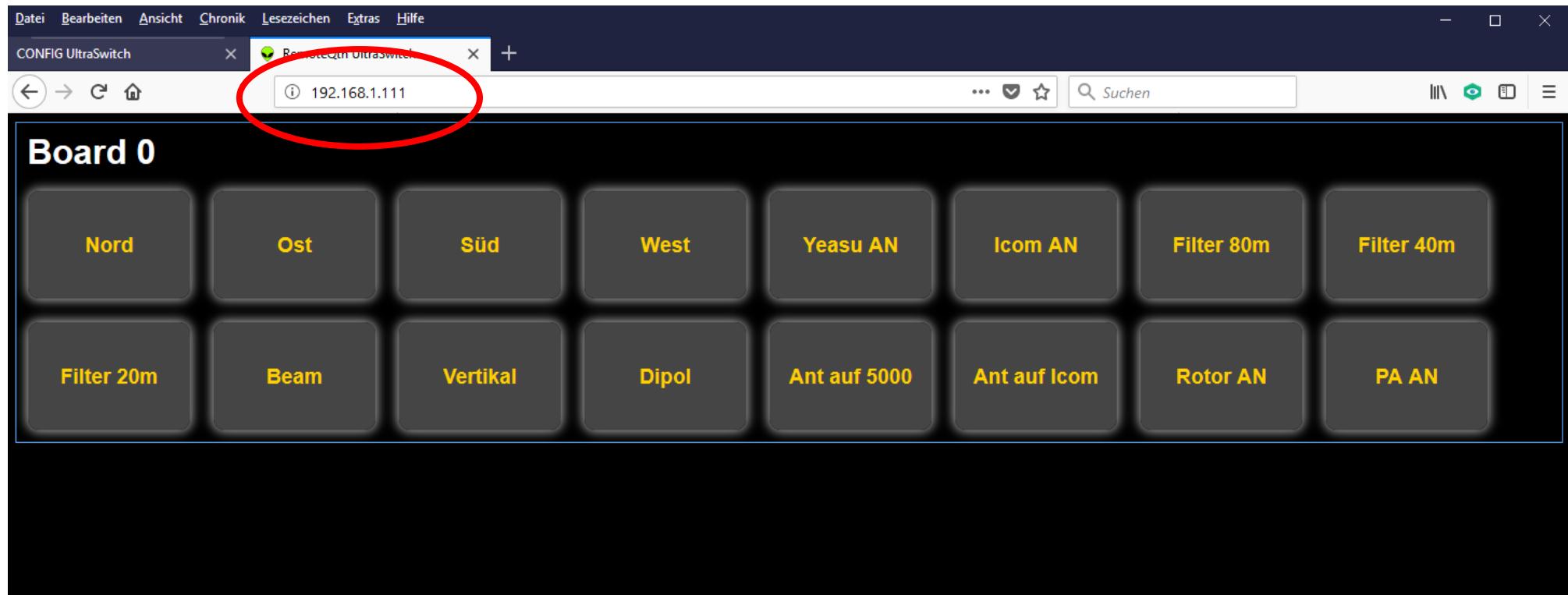
- Config.html im Verzeichnis aufrufen
- Wichtig: Derzeitige IP des Ultraswitch!
- Wichtig: Gewünscht IP des Ultraswitch!
- Restliche Information
- Submit!



BE CAREFUL WHAT YOU DO!! DOUBLECHECK TWICE BEFORE CLICKING SUBMIT :P



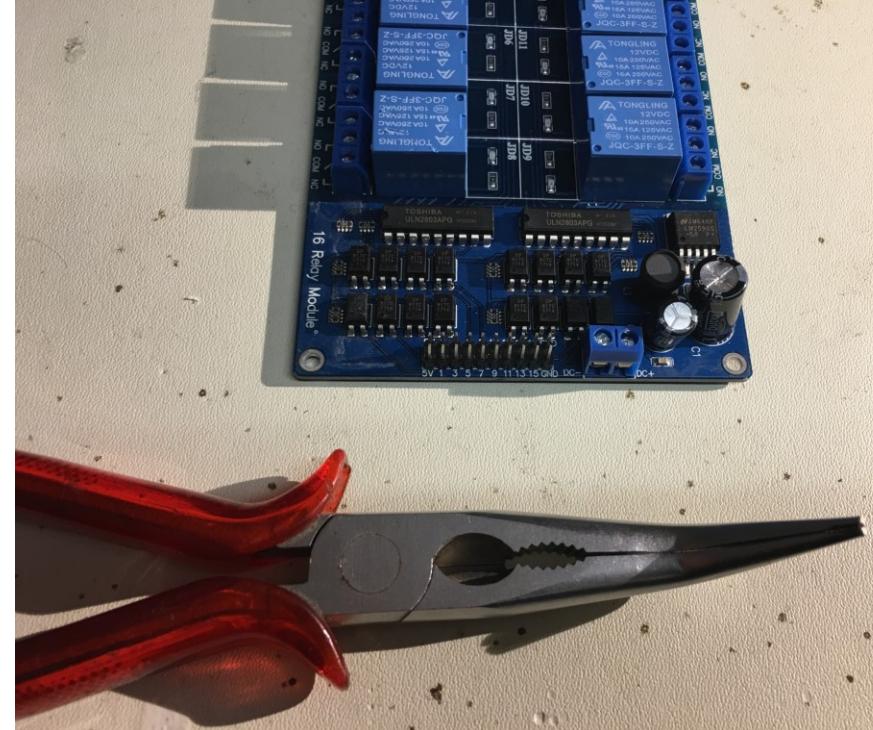
Finale IP-Adresse des Ultraswitch: Die UI



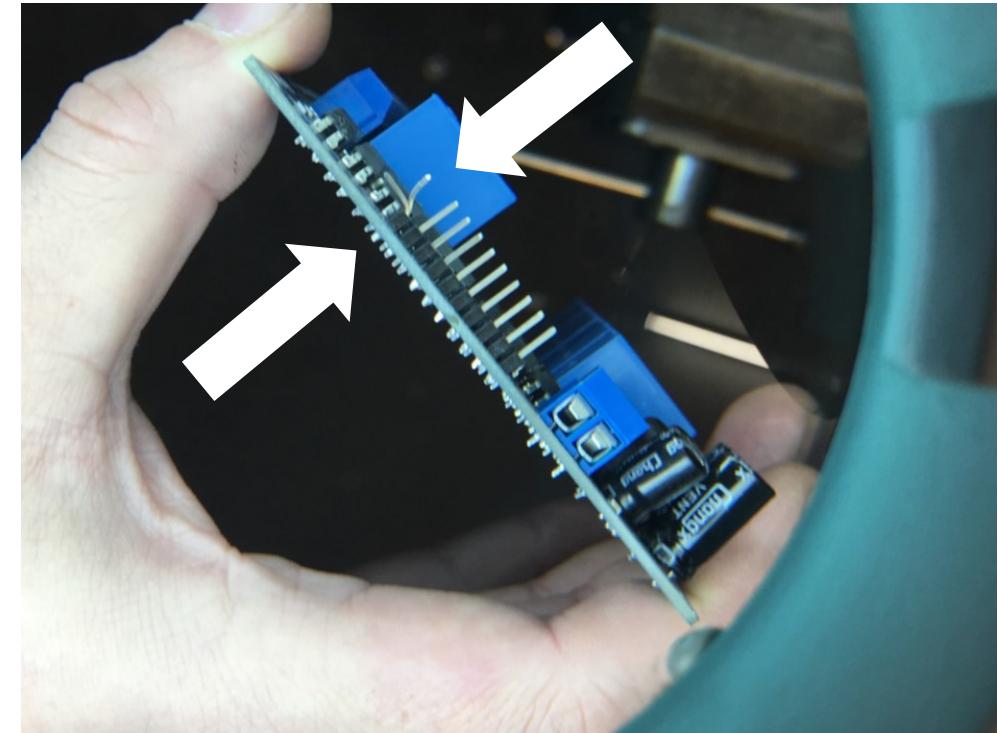
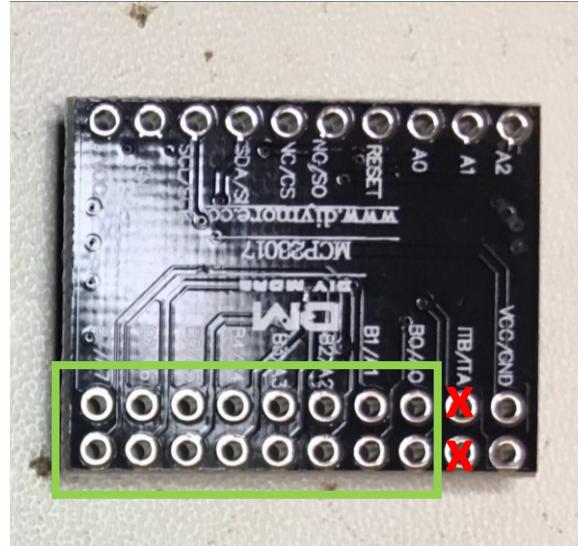
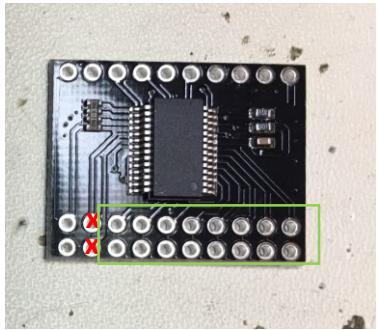
Das Relaysboard verkabeln... 1

Mit einer flachen Zange müssen die Pins zurecht gebogen werden, damit das MCP-Board auf den Header an der richtigen Stelle sitzt.

2 Pins (3,4) am MCP-Board müssen dabei frei bleiben... Ein nicht ganz leichtes Unterfangen...

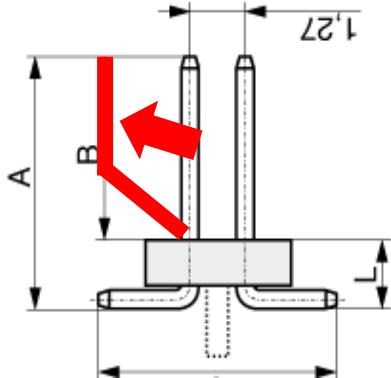


Das Relaysboard verkabeln... 2

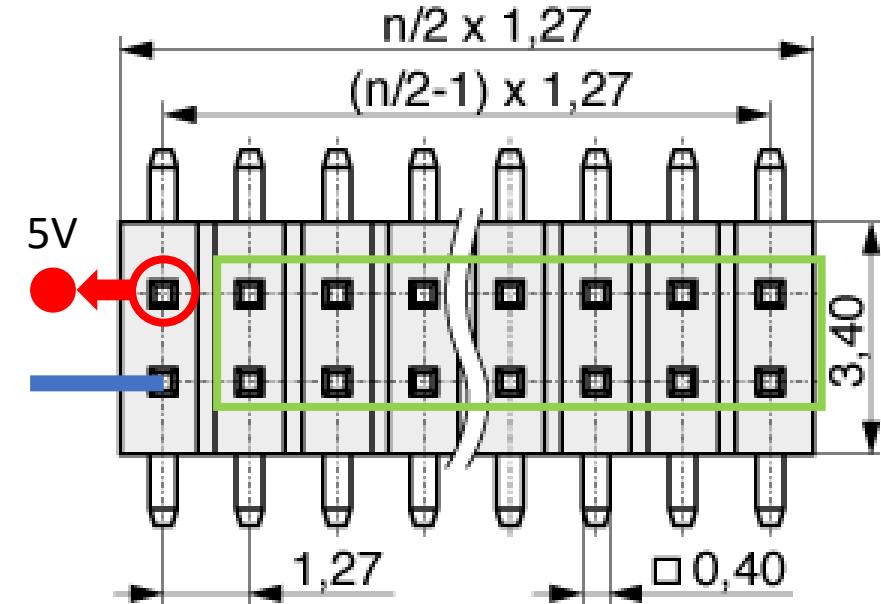
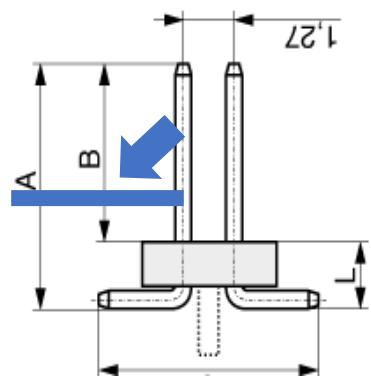


Das Relaysboard verkabeln... 3

- Pin 1: „Biegen“ um
1-Raster nach links!



- Pin 2:
Wird ganz umgebogen!



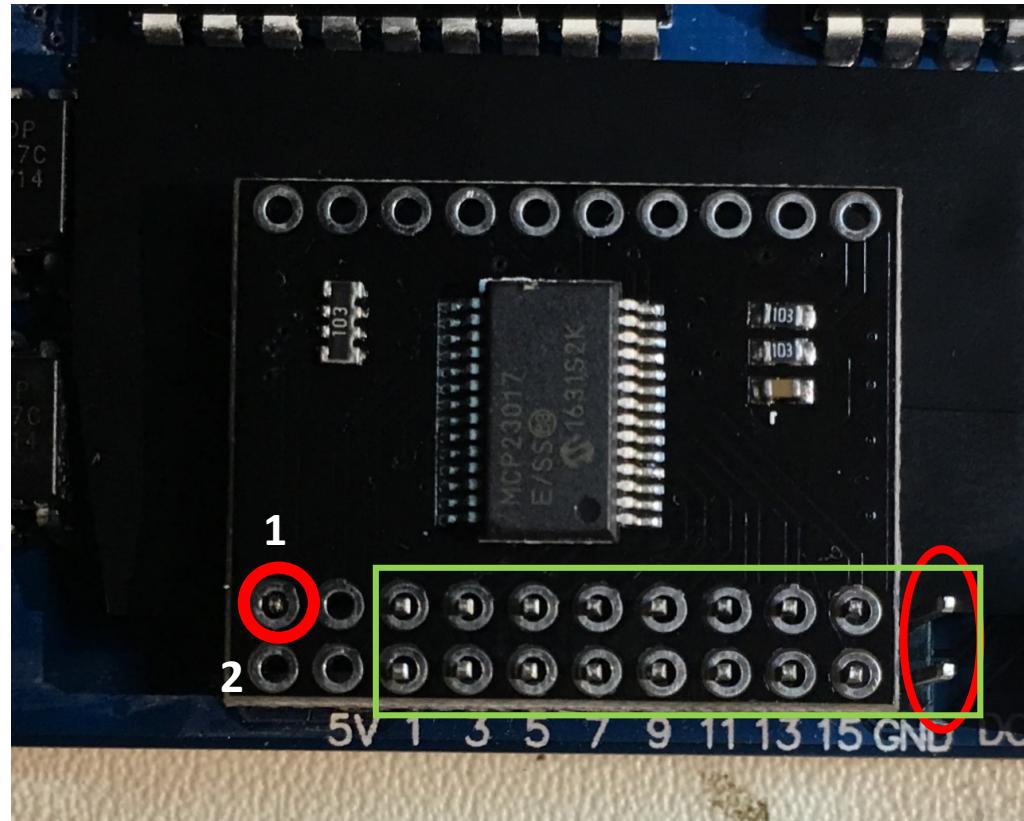
(((())))

Das Relayboard verkabeln... 4

Nun wird das MCP-Board auf die vorbereiteten Headerpins gesteckt:

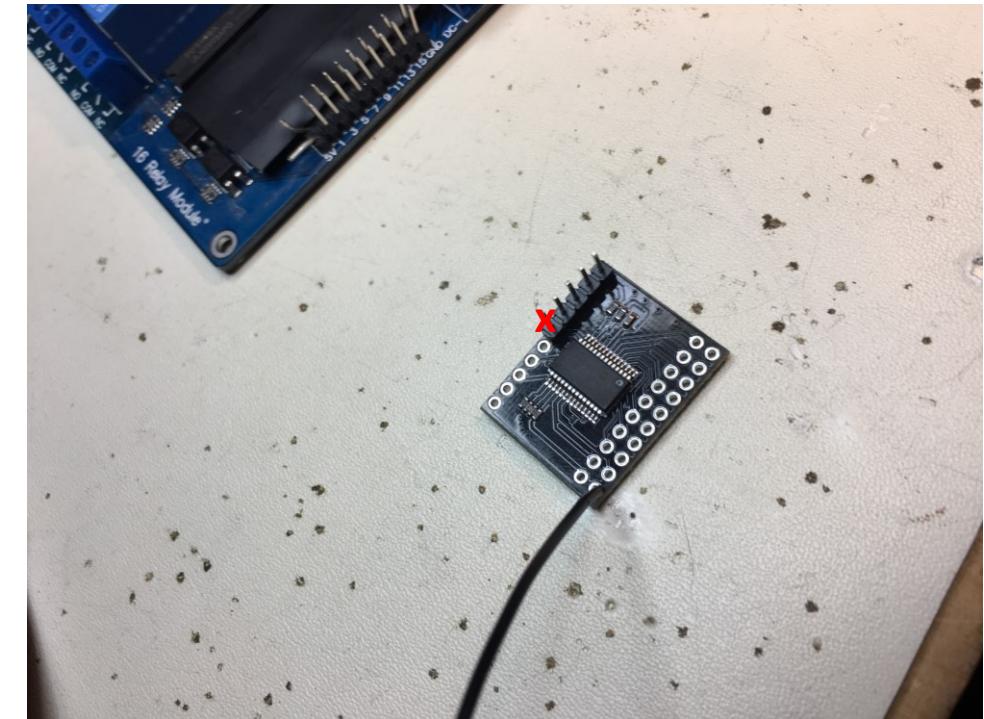
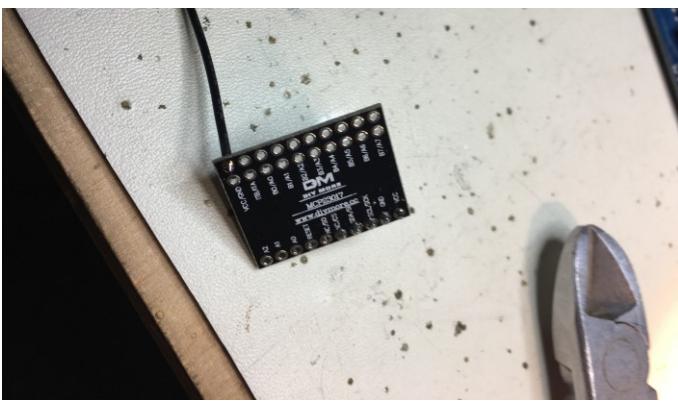
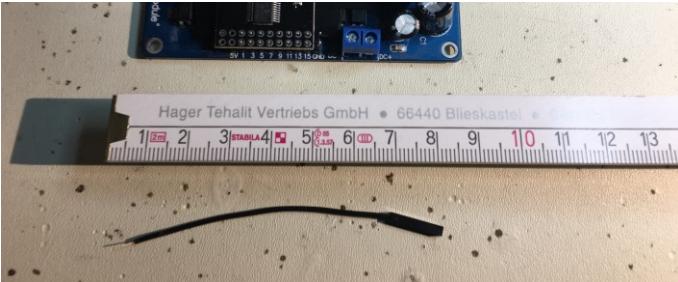
- In -1- sieht man den zum „S“ gebogenen Pin (wird später gelötet).
- Bei -2- ist darunter der komplett umgebogene Pin (nicht zu sehen)
- Stelle mit Pins ¾ ist frei.
- Ganz rechts: Freie GND-Pins (siehe später)

WICHTIG: Noch nicht festlöten!



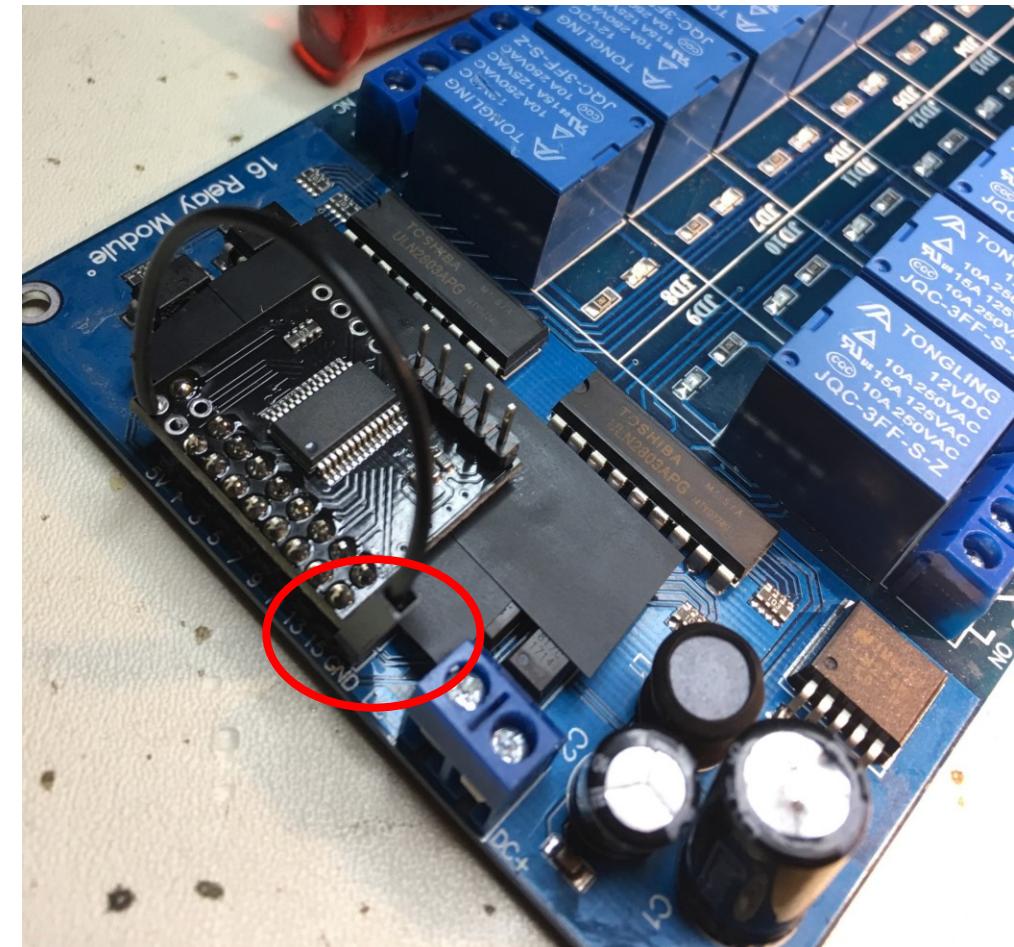
Das MCP-23017 verbinden

4 Pin-Header (4 reichen :P) und 1x GND-Wire schwarz verlöten:



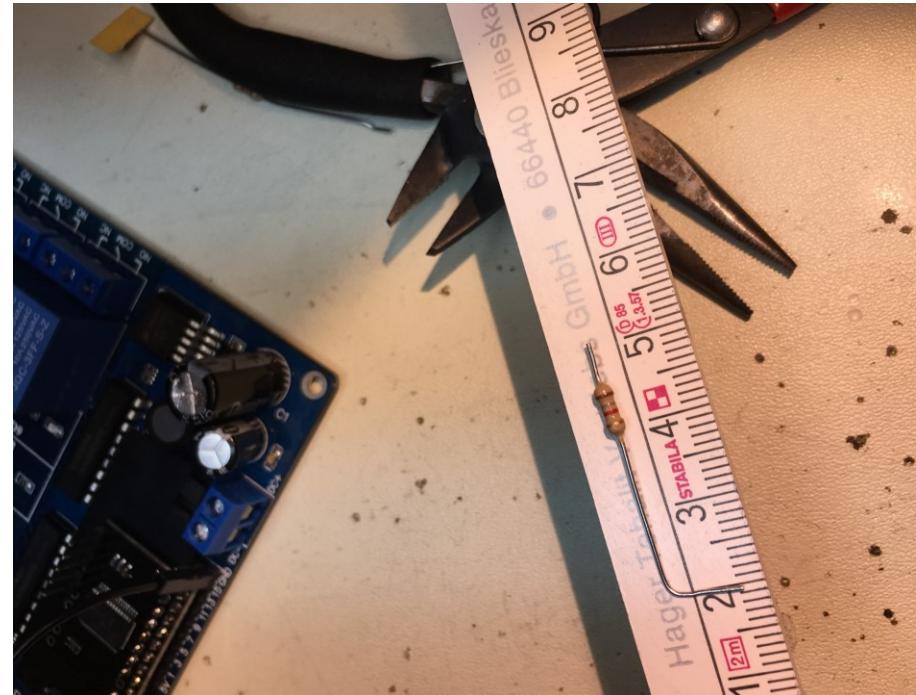
MCP-23017 auf das RelayBoard löten

- Voraussichtl. Auflagefläche mit Isolierband bekleben
- MCP-Board richtig platzieren
- MCP-PCB mit Relayboard verlöten
- Schwarzen GND-Wire mit RelayBoard verbinden



Den ESP-12F bestücken 1

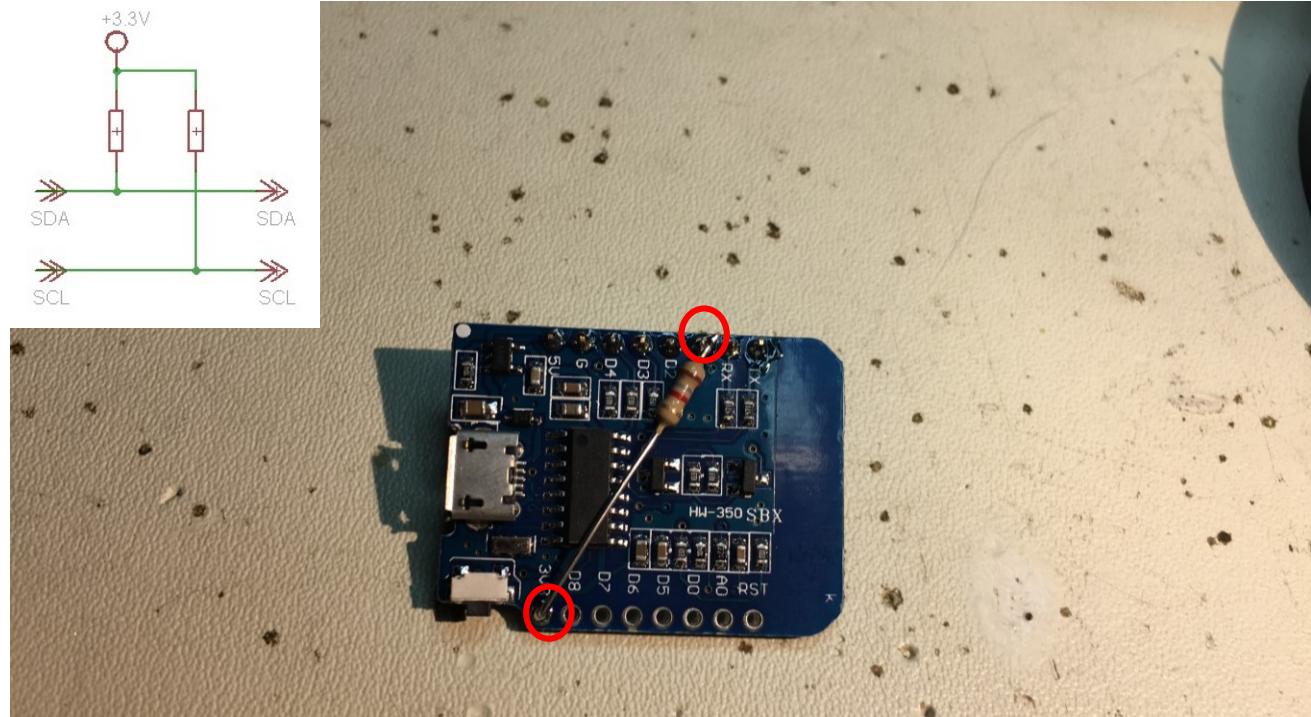
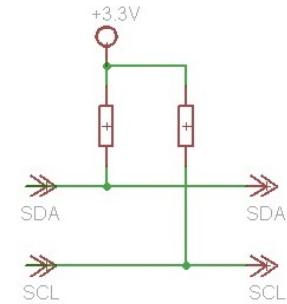
- Female Header anlöten & 1. 1k8 Widerstand vorbereiten:



Den ESP-12F bestücken 2

- 1. 1k8 Widerstand anlöten:

Pin D1 → 3V3!



- Achtung: Draht darf keine anderen Kontakte berühren...

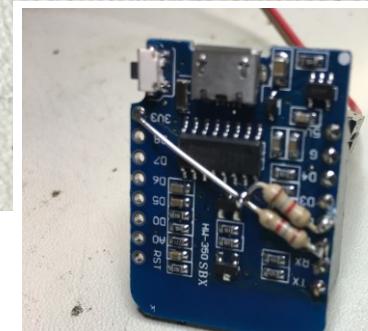
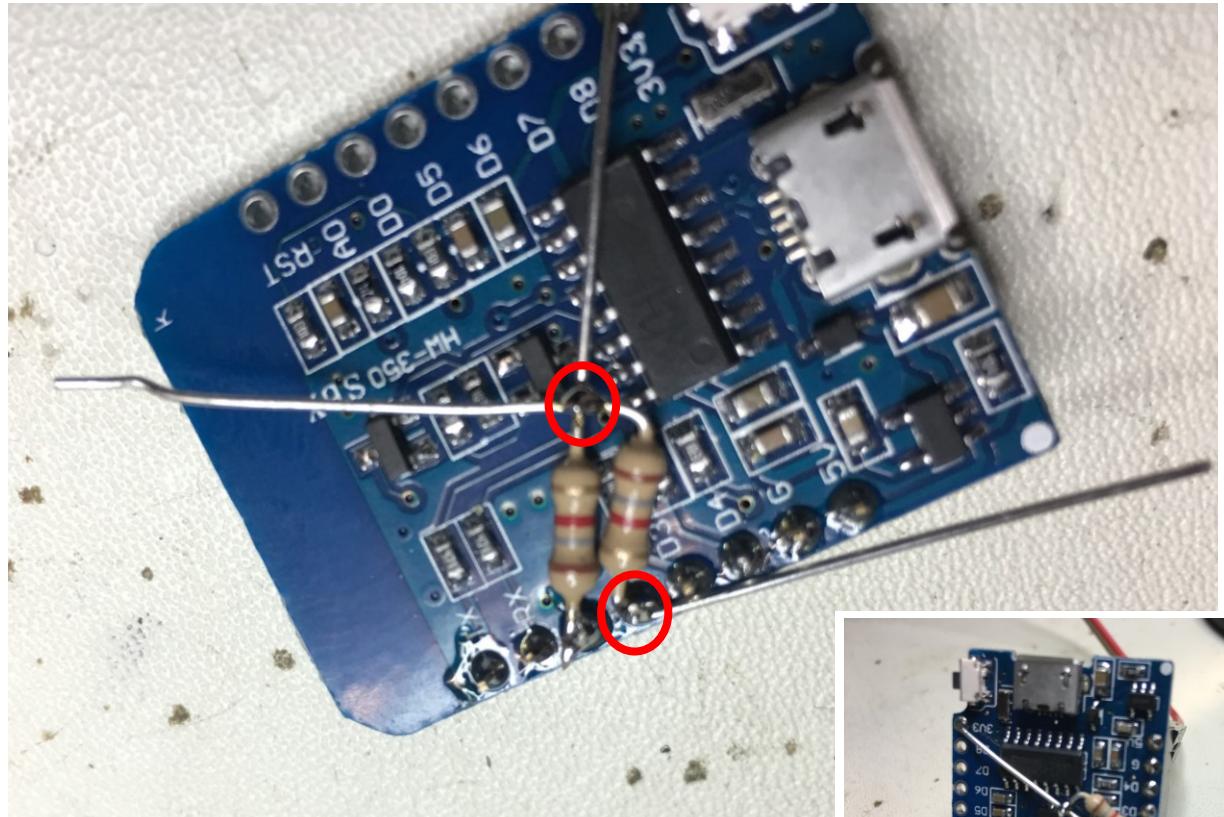


Den ESP-12F bestücken 3

- 2. 1k8 Widerstand zu einem „S“ – biegen und verlöten.

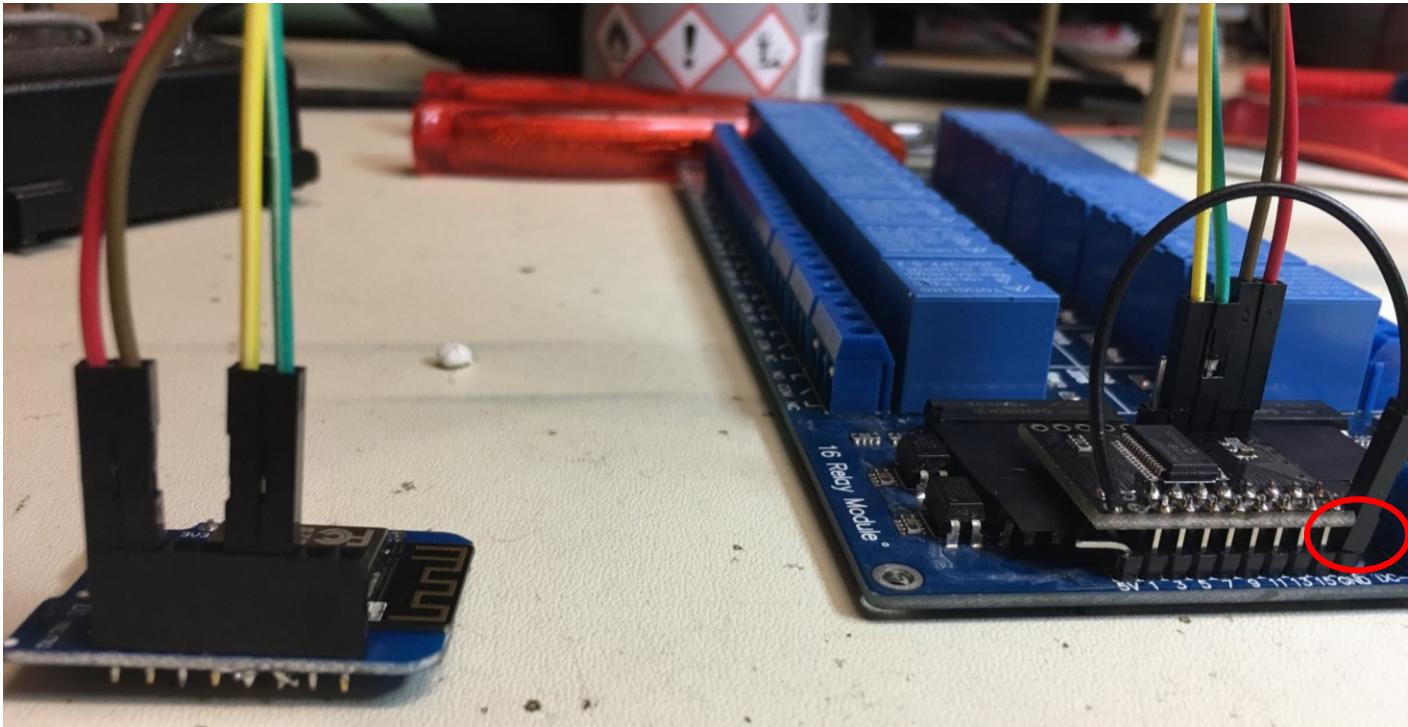
D2 → 1.Widerstand 3V3

- Abschneiden.
- Achtung: Draht darf keine anderen Kontakte berühren...



Alle drei zusammen...

- Am Wemos von links:
5V|GND|leer|leer|GELB|GRÜN
- Am MCP von links:
GELB|GRÜN|GND|5V
- SCHWARZ im freien GND (Relayb.)

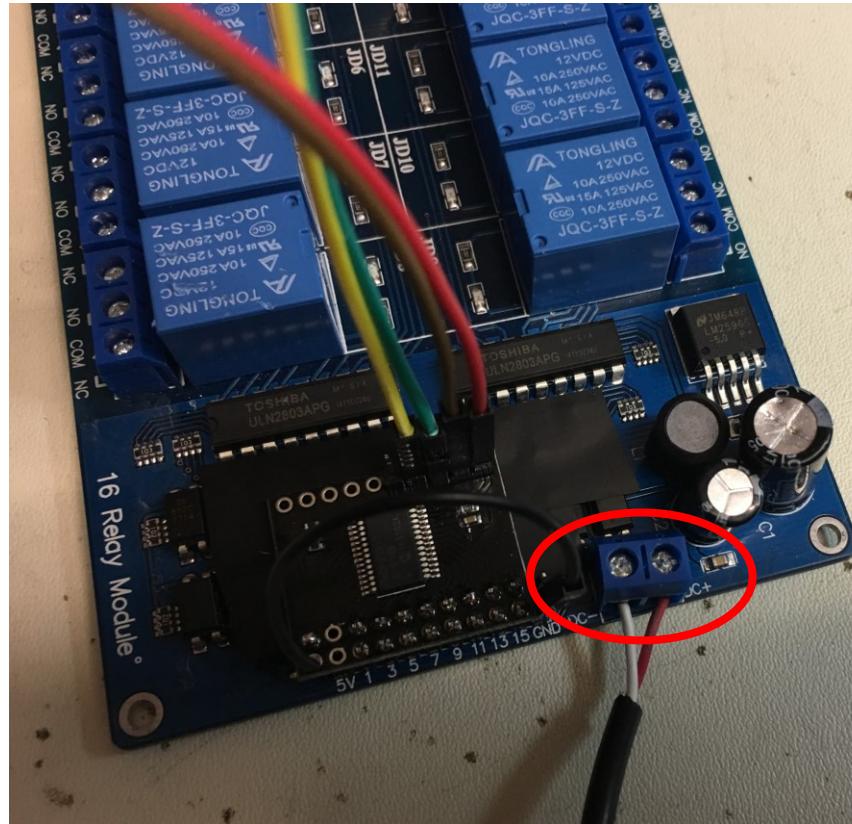
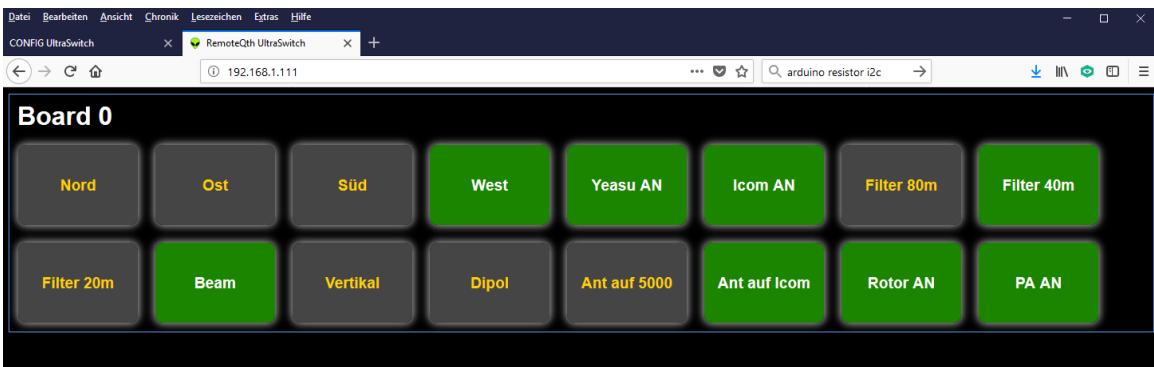


PS: 4 Pins reichen (der 5. war ein Versehen :P)



Am Ende...

- 12V verkabeln und anklemmen. (POLUNG!)
- Netzteil einstecken
- Browser starten



Customizing => Labels, Gruppierung, RelaysPerBoard

Einstellungsmöglichkeiten in der ultraswitch_content_v1.js:

- Labels => textboard0 = Labels für das 1. Board usw...

```
var textboard0 = ["Nord","Ost","Süd","West","Yerasu AN","Icom AN","Filter 80m","Filter 40m","Filter  
20m","Beam","Vertikal","Dipol","Ant auf 5000","Ant auf Icom","Rotor AN","PA AN"];      → Fasse dich kurz!!!
```

- Gruppen => board0Group = Gruppendefinition für das 1. Board usw...

```
var board0Group = [[0,1,2,3],[6,7,8],[9,10,11],[12,13]];
```

Erklärung: Auf dem 1. Board Nummer 0 gibt es 4 Gruppierungen. In der 1. Gruppierung ist Knopf0,1,2,3 miteinander verbunden. Es kann nur immer einer in der Gruppe "AN" sein. Usw...

- numberOfRelaysPerBoard = Wie viele Relays pro Board (1., 2., 3.,...)

```
var numberOfRelaysPerBoard = [14,16,16,16,16,16,16,16,16,16,16,16,16,16,16,16];
```

Erklärung: Erstes Board hat 14 Relays, 2. 16, (())



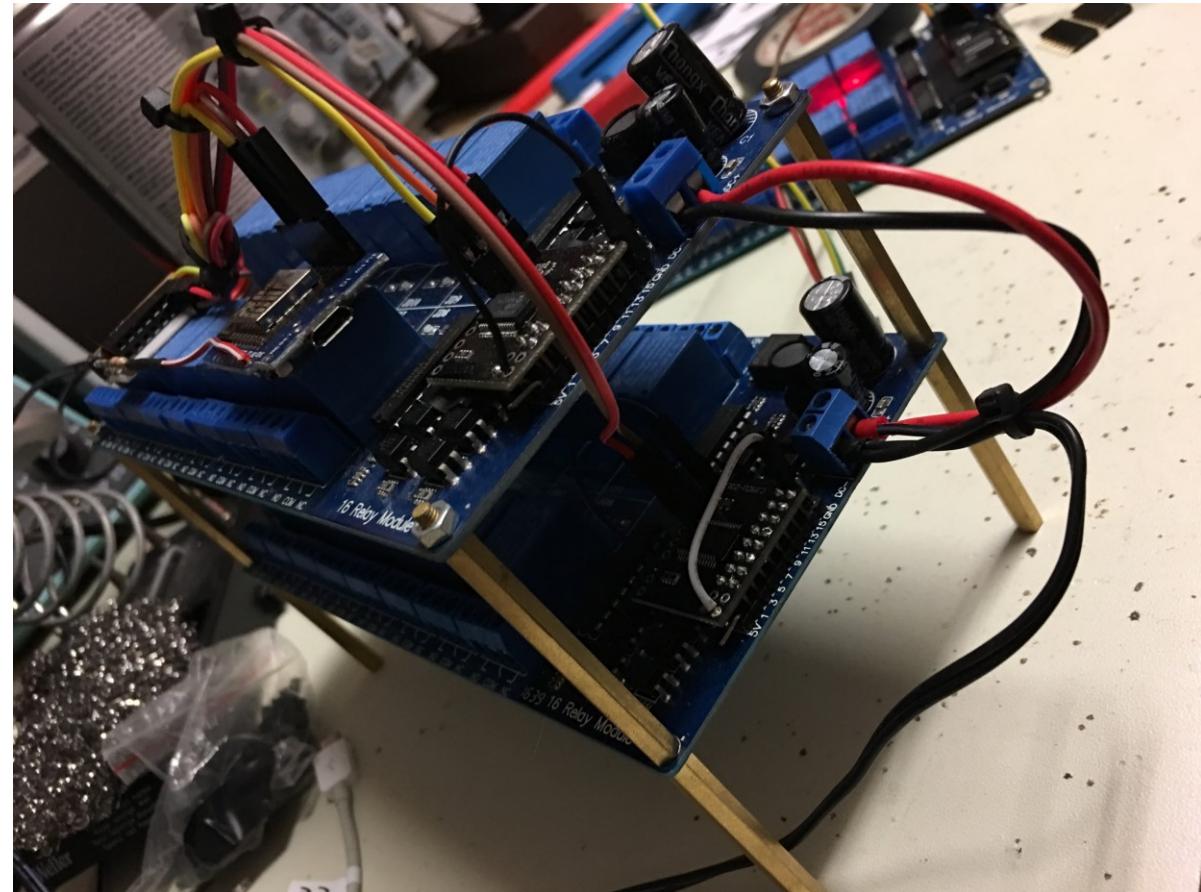
Aufbau und Funktionsweise des Sourcecodes

- Main-Sketch auf dem ESP12F für Hardwarekontrolle, JSON, WebServer:
UltraSwitch.ino
- Benutzte Bibliotheken:
Adafruit-MCP23017 → I2C-Kontrolle MCP23017
AdruinoJson → JSON Format für das Webserver-Response
WiFiManager → Einfach Konfiguration des Webservers
- Ultraswitch_content_v1.js → Konfiguration der Labes, Gruppen, etc.
Ultraswitch_v1.js → Schaltlogik und Steuerung der UI-Element
Ultraswitch_v1.css → UI Look&Feel, Buttongröße, Farben, etc.
- Setup_esp.html → Vorkonfiguration des ESP12F Access-Points
Config.html → Konfiguration der IP- und Servereigenschaften
index.html → Lokale ESP12F Anwendung (Einstiegspunkt)



Weitere Module...

- Weitere RelayBoards wie beschrieben zusammenbauen
- I2C-Adresse vergeben!
(nächste Folie)
- Nur Gelb|Grün mit Gelb|Grün verbinden (Parallel-Bus)
- Stromversorgung einfügen
- Config.html → Board-Anzahl
- Fertig!

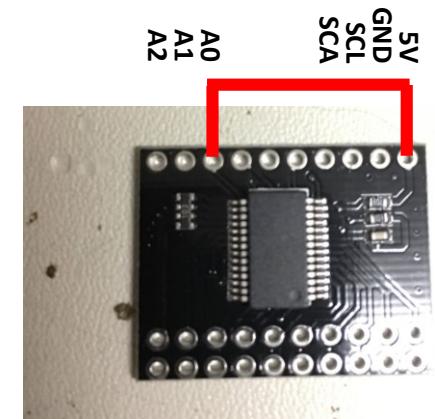
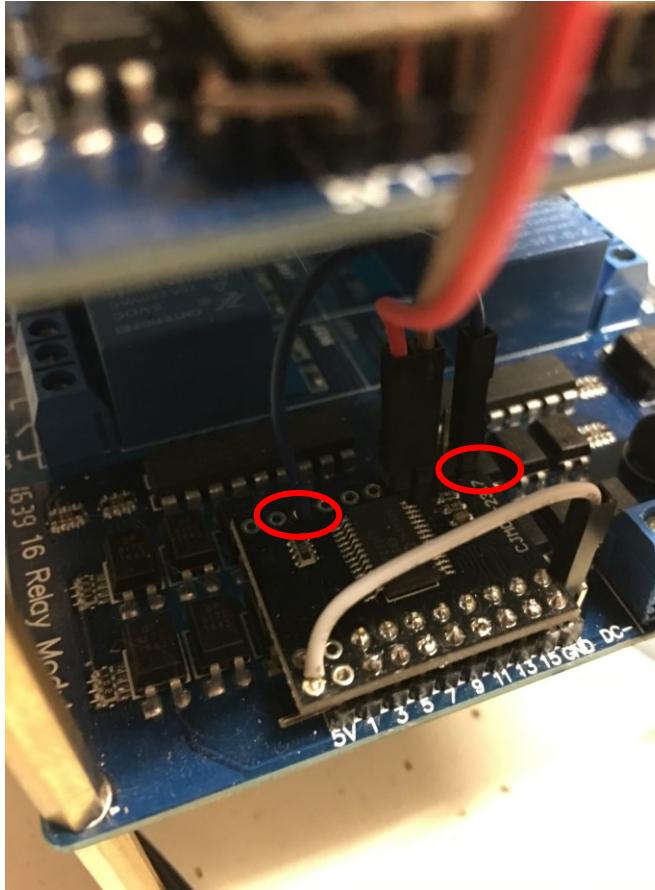


Weitere Module... I2C Adressierung => Bis zu 8 Devices!

- Die Verbindung mit 5V (HIGH) muss genau in der Reihenfolge durchgeführt werden:
0 => 1. Board, 1 => 2. Board,
2=> 3. Board, ...!!!

```
addr 0 = A2 low , A1 low , A0 low  000
addr 1 = A2 low , A1 low , A0 high 001
addr 2 = A2 low , A1 high , A0 low  010
addr 3 = A2 low , A1 high , A0 high 011
addr 4 = A2 high , A1 low , A0 low  100
addr 5 = A2 high , A1 low , A0 high 101
addr 6 = A2 high , A1 high , A0 low  110
addr 7 = A2 high, A1 high, A0 high 111
```

Hinweis: Nicht verbunden => LOW



Viel Platz für eigene Ideen...

- Mehrere Ultraswitch-Cluster
- Nutzung des ADC-IN für Messungen (Temperatur, etc.)
- Verwendung als Antennenschalter (Wichtig: Abschirmung, Gehäuse...)
- Zeitsteuerung von Pins (siehe mega-webswitch)
- Integration in N1NN (M. Hammelmann, DK5AX), Linuxanwendung
- Remote-Schalter für Remote Station (VPN wird empfohlen)
- Integration mit einem Raspberry PI als Server (UDP, Broadcast)
- ...



THE END

- Danke für Eure Aufmerksamkeit -
- Email: dm5xx@gmx.de -

